ARMY RESEARCH LABORATORY

Linear Viscoelastic Predictions Using a Single-Chain Mean-field Discrete Slip-link Model

by Tanya L. Chantawansri, Yelena R. Sliozberg, and Jan W. Andzelm

ARL-TR-5295 **August 2010**

NOTICES

Disclaimers

The findings in this report are not to be construed as an official Department of the Army position unless so designated by other authorized documents.

Citation of manufacturer's or trade names does not constitute an official endorsement or approval of the use thereof.

Destroy this report when it is no longer needed. Do not return it to the originator.

Army Research Laboratory
Aberdeen Proving Ground, MD 21005-5069

ARL-TR-5295 **August 2010**

Linear Viscoelastic Predictions Using a Single-Chain Mean-field Discrete Slip-link Model

Tanya L. Chantawansri, Yelena R. Sliozberg, and Jan W. Andzelm
Weapons and Materials Research Directorate, ARL

REPORT DOCUMENTATION PAGE

Form Approved
OMB No. 0704-0188

1. REPORT DATE *(DD-MM-YYYY)*	2. REPORT TYPE	3. DATES COVERED (From - To)
August 2010		

4. TITLE AND SUBTITLE
Linear Viscoelastic Predictions Using a Single-Chain Mean-field Discrete Slip-link Model

5a. CONTRACT NUMBER

5b. GRANT NUMBER

5c. PROGRAM ELEMENT NUMBER

6. AUTHOR(S)
Tanya L. Chantawansri, Yelena R. Sliozberg, and Jan W. Andzelm

5d. PROJECT NUMBER

5e. TASK NUMBER

5f. WORK UNIT NUMBER

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)
U.S. Army Research Laboratory
ATTN: RDRL-WMM-G
Aberdeen Proving Ground, MD 21005-5069

8. PERFORMING ORGANIZATION REPORT NUMBER
ARL-TR-5295

9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)

10. SPONSOR/MONITOR'S ACRONYM(S)

11. SPONSOR/MONITOR'S REPORT NUMBER(S)

12. DISTRIBUTION/AVAILABILITY STATEMENT
Approved for public release; distribution unlimited.

13. SUPPLEMENTARY NOTES

14. ABSTRACT
Currently, there is substantial interest in predicting linear viscoelasticity data through the use of computational modeling. Tube models have been long used for this calculation, but limitations with the model have spurred research in different computational techniques. One such theory is the self-consistent single chain mean-field discrete slip-link model developed by Schieber and coworkers. This theory has been programmed by Schieber and coworkers into a FORTRAN90 code, and has been made available to the U.S. Army Research Laboratory. In this report, we give an overview of the theory, as well as an introduction on how to use this program. We use this program to calculate the storage and loss modulus for polystyrene. This data is compared to experimental results.

15. SUBJECT TERMS
Viscoelastic properties, slip-link model, computational modeling

16. SECURITY CLASSIFICATION OF:			17. LIMITATION OF ABSTRACT	18. NUMBER OF PAGES	19a. NAME OF RESPONSIBLE PERSON
a. REPORT	b. ABSTRACT	c. THIS PAGE			Tanya L. Chantawansri
Unclassified	Unclassified	Unclassified	UU	36	19b. TELEPHONE NUMBER (*Include area code*) (410) 306-2777

Contents

List of Figures

List of Tables

Acknowledgments

We would like to thank Professor Jay D. Schieber for providing the program and Renat N. Khaliullin and Marat Andreev for useful discussions. This research was supported in part by an appointment to the Postgraduate Research Participation Program at the U.S. Army Research Laboratory (ARL) administered by the Oak Ridge Institute for Science and Education through an interagency agreement between the U.S. Department of Energy and ARL.

INTENTIONALLY LEFT BLANK.

1. Background

Since polymeric materials are dense ensembles of large molecules formed by linking together many repeat units though covalent bonds, the interaction between the polymer chains leads to chain entanglements. To describe the entangled state, several tube models have been developed to quantify the dynamics arising due to the entanglements. These models predict linear viscoelasticity for linear entangled polymers with very good agreement with experimental data. The shortcoming of these tube models is that they are restricted to linear systems and cannot be generalized for bidisperse polymers, branched systems, or cross-linked networks without fundamental modifications. In addition, most existing tube models lack agreement with small-amplitude oscillatory shear flow experiments of bidisperse linear melts. To overcome this shortcoming, a newer discrete slip-link model (DSM) has been developed.

The slip-link concept was first introduced by Doi and Edwards in 1978 (*3*) and was further developed by Schieber into the DSM (*9, 5, 10, 7*). The DSM is a general approach, which allows for the prediction of both the linear rheological behavior as well as the nonlinear flow of entangled polymers without additional parameters. The model chain also exhibits primitive-path-length fluctuations and chain stretching, which allows it to be applied to nonlinear deformations. In the current version of the DSM used in this report, constant chain friction and constraint release are also considered to model the creation and destruction of entanglements. Unlike former slip-link models, this method is capable of modeling both polymer melts and solutions, and could be generalized for bidisperse, branched, and cross-linked polymers as well as for biological semi-flexible polymers (f-actins or intermediate filaments).

The DSM is a mean-field simulation approach, capable of describing a system composed of entangled polymers of arbitrary architecture including branched or cross-linked systems. To run a DSM simulation, three input parameters are required: the number of Kuhn segments, N_k, and two fitting parameters, β and τ_k. The number of Kuhn segments is fully determined from the molecular weight of the chain, while the fitting parameter, τ_k, is related to the average time for a Kuhn step to jump through a given entanglement. The parameter, β, is derived from the entanglement spacing on a chain. These parameters are described in greater detail in section 2.1.

In the slip-link model, the chains are described as a random walk, which are valid for polymeric chains with contour length and entanglement spacing longer then several Kuhn steps (figure 1). Each chain has a constant number of Kuhn segments, where each Kuhn segment has a constant length, a_k. Since a random walk of N_i Kuhn steps between two entanglements with a connector vector, $\mathcal{Q}$, has a Gaussian conditional distribution, the free

energy of an entangled i^{th} strand of a chain is given by

$$\frac{F_s(\mathcal{Q}_i, N_i)}{k_B T} = \frac{3\mathcal{Q}_i^2}{2N_i a_k^2} + \frac{3}{2}\ln\left[\frac{2\pi N_i a_k^2}{3}\right], \tag{1}$$

where k_B is the Boltzmann constant and T is the temperature. The free energy of a chain composed of Z strands is a sum of the free energies associated with the entangled strands:

$$F(\Omega) = \sum_{i=2}^{Z-1} F_s(\mathcal{Q}_i, N_i), \tag{2}$$

where Ω is the chain conformation. The free energy of the dangling ends ($i = 1$ and $i = Z$) is a constant, which can be set to zero since it only contributes as a constant shift in the total free energy. Although the total number of Kuhn steps (N_k) in a chain is constant, the number of Kuhn step in a strand, N_i, fluctuates between neighboring strands through Brownian forces and free energy differences. In DSM, the number of Kuhn steps shifted from one strand to another is an integer number where only one Kuhn step can be shuffled at a time through entanglement i. The entanglements can be destroyed or created by sliding dynamics (SD) at the end of the chains or by constrain release (constraint dynamics [CD]) in the middle of the chain. The entanglement is destroyed by SD when a dangling end is abandoned by the last Kuhn step, while the creation process is fully determined from detailed balance. When one entanglement is destroyed by CD, another is destroyed by SD automatically. The equilibrium distribution of such a chain is given by the modified Maxwell-Boltzmann relation

$$p_{eq}(\Omega) = \frac{\delta(N_K, \sum_{i=1}^{Z} N_i)}{J} \exp\left[-\frac{F(\Omega)}{k_B T}\right] \exp\left[\frac{\mu^E(Z-1)}{k_B T}\right] \prod_{i=1}^{Z-1} p^{CD}(\tau_i^{CD}), \tag{3}$$

where $p^{CD}(\tau_i^{CD})$ is the probability density for the i^{th} entanglement with a characteristic CD lifetime, τ^{CD}. The Kronecker delta function $\delta(i, j) = \delta_{i,j}$ is responsible for the conservation of Kuhn steps in a chain and J is a normalization constant. The entanglement chemical potential of the surrounding chains, μ_E, is responsible for fluctuations in the number of entanglements on the chain.

Using the DSM, the relaxation modulus, $G(t)$, can be predicted using the Green-Kubo expression:

$$G(t) = \frac{1}{n k_B T} < \tau(0)\tau(t) >_{eq}, \tag{4}$$

where the stress tensor $\tau(t)$ is defined as

$$\tau(t) = -n \left\langle \sum_{j=2}^{Z-1} \mathcal{Q}_j \left[\frac{\partial F[\Omega]}{\partial \mathcal{Q}_j}\right] \right\rangle_{eq}, \tag{5}$$

where n is the number density of polymer chains and $<>_{eq}$ is an ensemble average.

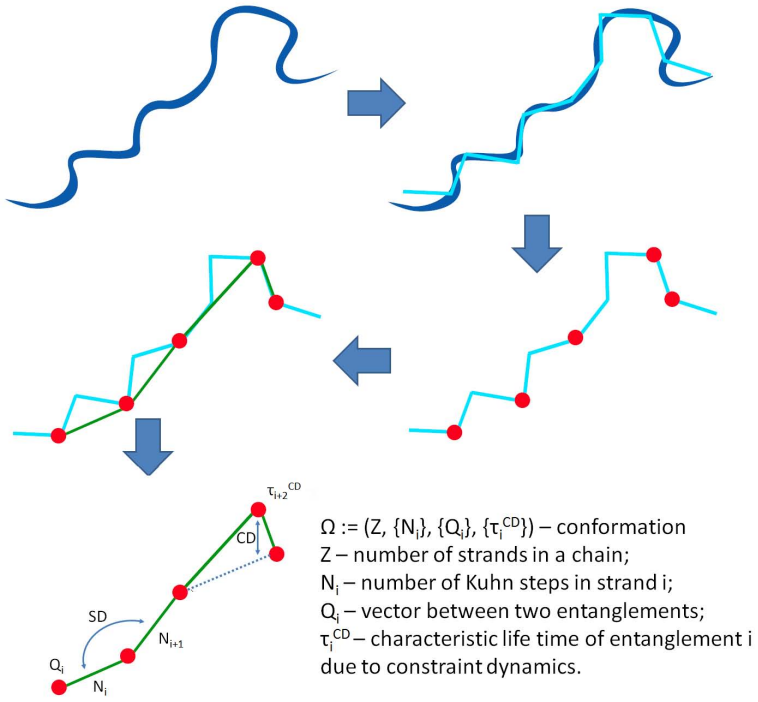

Figure 1. The schematic of the polymer for modeling linear polymers using DSM. A polymer (shown in navy) in a mean field is first discretized using the discrete Gaussian chain model (shown in cyan). The number and location of entanglements (shown in red) are then determined by equation 3. The number of Kuhn steps in strand i, N_i, is then determined by connecting two entanglements, i and $i+1$, with the orientation vector Q_i (shown in green).

Since experimental data related to the relaxation modulus are typically presented in the frequency domain ($G^* := i\omega[F][G(t)]$), it is necessary to transform the data into the frequency domain. This is done by fitting the simulated relaxation modulus with a continuous curve, which then transformed into the frequency domain using the analytical Fourier transform. Thus the complex relaxation modulus G^* is obtained by multiplying the Fourier transform of $G(t)$ with $i\omega$.

From the relaxation modulus, the storage $G'(\omega)$ and loss $G''(\omega)$ modulus can be calculated. This is performed by calculating the continuous spectrum of the relaxation times, $h(\tau)$, introduced in the following expression for the relaxation modulus

$$G(t) = G_N^0 \int_0^\infty d\tau \frac{h(\tau)}{\tau} \exp\left(\frac{t}{\tau}\right), \tag{6}$$

where it can be decomposed with the power-law spectrum proposed by Baumgaertel,

Schausberger, and Winter

$$h(\tau) = \frac{\sum_{i=1}^{m} \tau^{\alpha_i} H(\tau_{i-1} - \tau) H(\tau - \tau_i) \prod_{j=1}^{i-1} \tau_j^{\alpha_j - \alpha j+1}}{\sum_{i=1} m \frac{\tau_i^{\alpha_i} - \tau_{i-1}^{\alpha_i}}{\alpha_i} \prod_{j=1}^{i-1} \tau_j^{\alpha_j - \alpha_{j+1}}}. \tag{7}$$

In equations 6 and 7, G_N^0 is the plateau modulus, m, is the number of modes; τ_i are time constants; and α_i are power-law exponentials.

These coefficients can then be used to calculate the storage and loss modulus

$$G'(\omega) = G_N^0 \omega^2 \sum_{i=1}^{m} \frac{\prod_{j=0}^{i-1} \tau_j^{\alpha_j - \alpha_{j+1}}}{\alpha_i + 2} \left[{}_2F_1\left(1, \frac{\alpha_i + 2}{2}; \frac{\alpha_i + 4}{2}; -\omega^2 \tau_i^2\right) \tau_i^{\alpha_i + 2} \right. \tag{8}$$
$$\left. - {}_2F_1\left(1, \frac{\alpha_i + 2}{2}; \frac{\alpha_i + 4}{2}; -\omega^2 \tau_{i-1}^2\right) \tau_{i-1}^{\alpha_i + 2} \right] / \sum_{i=1}^{m} \frac{\prod_{j=0}^{i-1} \tau_j^{\alpha_j - \alpha_{j+1}} (\tau_i^{\alpha_i} - \tau_{i-1}^{\alpha_i})}{\alpha_i},$$

and

$$G''(\omega) = G_N^0 \omega \sum_{i=1}^{m} \frac{\prod_{j=0}^{i-1} \tau_j^{\alpha_j - \alpha_{j+1}}}{\alpha_i + 2} \left[{}_2F_1\left(1, \frac{\alpha_i + 1}{2}; \frac{\alpha_i + 3}{2}; -\omega^2 \tau_i^2\right) \tau_i^{\alpha_i + 1} \right. \tag{9}$$
$$\left. - {}_2F_1\left(1, \frac{\alpha_i + 1}{2}; \frac{\alpha_i + 3}{2}; -\omega^2 \tau_{i-1}^2\right) \tau_{i-1}^{\alpha_i + 1} \right] / \sum_{i=1}^{m} \frac{\prod_{j=0}^{i-1} \tau_j^{\alpha_j - \alpha_{j+1}} (\tau_i^{\alpha_i} - \tau_{i-1}^{\alpha_i})}{\alpha_i},$$

respectively, where ${}_2F_1(a, b; c; d)$ represents the hypergeometric function.

2. Overview

A FORTRAN90 DSM program for linear monodisperse entangled polymer melts was provided by Professor Schieber from the Illinois Institute of Technology. Using this program, the simulated relaxation spectrum can be calculated. This data are then exported into Mathematica where the data can be fitted into a continuous curve. From this curve, the shear and loss modulus can be subsequently calculated.

To run this program, two parameters are necessary: the number of Kuhn steps, N_k, and the model parameter, β. The model parameter, τ_k, is not needed until the calculation of the storage and loss modulus. The code is provided on both the mjm and harold supercomputers run by the U.S. Army Research Laboratory at the DoD Supercomputing Resource Center.

2.1 Choice of Parameters

2.1.1 The Number of Kuhn Steps, N_k

The input parameter, N_k, is fully determined from the molecular weight of the chain

$$N_k = M_w/M_k, \tag{10}$$

where M_w and M_k are molecular weight of polymer and Kuhn steps, respectively. M_w and M_k can be found from experiment and/or literature.

2.1.2 Model Parameter, β

The model parameter, β, is defined from

$$\beta = \exp\left[\frac{-\mu^E}{k_B T}\right], \tag{11}$$

where β depends on the polymer chemistry and solvent concentration. It is assumed to be independent of temperature, which is valid assumption as long as the chain flexibility does not change significantly. For long polymer ($N_k \to \infty$), β is estimated from the plateau modulus of the polymer, G_N^0:

$$\beta = \frac{\rho RT}{M_k G_N^0} - 1, \tag{12}$$

where ρ is the polymer density and R is the ideal gas constant. If the molecular weight of entanglement, M_e, is available instead of G_N^0, β is evaluated from

$$\beta = \frac{5}{4}\frac{M_e}{M_k} - 1. \tag{13}$$

This equation is a result of the following relationship:

$$M_e = \frac{4}{5}\frac{\rho RT}{G_N^0}. \tag{14}$$

The DSM does not describe the glassy regime in polymer dynamics and, therefore, the experimental value of G_N^0 (apparent plateau modulus) is lower than theoretical plateau modulus used for calculation of β. Thus, the initial value of β could be taken from equations 12 or 13 and its value could be later adjusted to the height of relaxation modulus ($G_N^0 = G(0)$) when calculating the loss and storage modulus, or the initial value of β could be taken to be smaller then the calculated value. For polymer solutions, M_e^{solution} could be estimated from the experimental data collected for melts as

$$M_e^{\text{solution}} = \frac{M_e^{\text{melt}}}{\phi^\alpha}, \tag{15}$$

where α depends on the solvent and is usually taken as $\alpha \approx 1.3$ and ϕ is the volume fraction of the polymer. If M_e and α are not known (no experimental data are available), they can be evaluated from atomistic simulation.

2.1.3 Model Parameter: τ_k

The parameter, τ_k, is a time constant related to the friction coefficient of a single step in the chain, which only depends on the chemistry and temperature of the polymer and solvent concentration (if present). The time dependence of the model prediction is also normalized by the time constant, τ_k, where its value can be determined from fitting simulation results to a single experimental data set (linear viscoelastic experiment). The adjustment of the time constant does not influence the shape of the relaxation modulus, but shifts it in the frequency domain.

2.2 Running the DSM Code

Three files are needed to run the DSM code: (1) the input file called input.dat, (2) the executable named slm_run, and (3) the job submitting script.

2.2.1 Location of DSM Code

The DSM code can be obtained off the mjm machine at /usr/people/yrs/General and the harold machine at /usr/people/tchantaw/SL_code.

2.2.2 Modifying the Input File: input.dat

The input.dat file takes the following form:

5.d0	!β (double)
55	!N_k: number of Kuhn steps (int)
1	!Number of chains (int)
0.d0 0.d0 0.d0 0.d0 0.d0 0.d0	!deformation tensor (xx,xy,xz,yy,yz,zz) (double)
0	!CD off(0)/on monodisperse(1)/on from file(2) (int)
0	!SD off(0)/on (int)
1	!G(t) estimation on/off (int)
0	!f_d(t) and destruction rate estimation on/off (int)
0	!Diffusion on/off (int)
1.d0	!time step (double)
100000000.d0	!T - simulation time
500000.d0	!saving interval period

The first two lines are the input parameters, β and N_k, mentioned in section 2.1. The third parameter is the number of chains in the ensemble, which is fixed to 1 since this is a single-chain mean-field theory. The deformation tensor is the transpose of the imposed velocity gradient, which is set equal to 0 for the calculation of the relaxation modulus. The fifth to ninth parameters are switches, where the value is set to 0 for on and 1 for off. For β values in the range $2 < \beta < 50$ and $3 < N_k/\beta < 80$, CD can be used to model constraint release where entanglements are created with a predetermined probability density. This probability density creates short-lived entanglements more often then long-lived ones since they are destroyed more often. If β and/or N_k/β fall out of this range, the f_d(t) and destruction rate estimation should instead be turned on. SD indicates sliding dynamics, and G(t) is turned on if the relaxation modulus is desired. The ninth parameter indicates if the diffusion coefficent should be calculated. The time step used in the code fluctuates within a simulation run, where the inputted value is used to determine how often the simulation results are written to a file. The simulation time is the total time that the simulation will run, and the saving interval period indicates the number of time steps between which each simulation state is saved.

The simulation time should be long enough to make the results reproducible. This means that the results of the longer simulation should be the same as the results obtained from a shorter simulation up to a small error. This error is not specified by Schieber's group. Note, that model works only for

$$2 < \frac{4}{5}\frac{M_e}{M_k} - 1 < 50. \tag{16}$$

2.2.3 Compiling the Code

The subrountines and functions related to the DSM program are shown in table 1. This list is taken from the readme file provided with the DSM code. Since this report is a brief guide, not all of these variables and their functions in the DSM code are defined, but more information regarding the variables can be found in reference 6.

After inputting the proper values into input.dat, compile the code using a provided script by running ./slm_mjm. This produces an executable called slm_run.

Although this code has only been tested using the Intel compiler, other FORTRAN90 compilers can be used. If a different FORTRAN90 compiler is desired, modify the script titled "slm" by replacing the word "fortran" with the name of the new compiler. Then to compile the code, just run the command **./slm**.

On mjm and harold, the FORTRAN code should be compiled using the Intel compiler options, -i8 and -O1. The option -i8 specifies the size for integer and logical variables to 64, while the -O1 option is an optimization for speed. This code does not generate proper

Table 1. Programs in DSM program.

conformations.f90	module; carries all the global variables for SLM code
conv_func.f90	function; gives a front-end position number
BD_step.f90	subroutine; changes conformation for SD per time step
BD_W_calc.f90	function; calculates $W_S D^i$ for Kuhn step shift to the left or right
bico2.f90	function; calculates binomial coefficient from Numerical Recipes
bico.f90	function; calculates ratio of two binomial coefficients
bnldev.f90	function; used for bico2.f90 from Numerical Recipes
main.f90	main file; reads files: input.dat and CD_spectr.dat; saves simulation variables in cha_♯.dat; copies some files to resolve access issue; generates file names
CR_step.f90	subroutine; changes conformation for CD per time step
dbd_dcr_plot.f90	subroutine; plots SD destruction and CD destruction
Diff_copy.f90	subroutine; copies Diff_♯.dat to Diff2_♯.dat
Diff_plot.f90	subroutine; plots diffusion
dtfunc.f90	finds timestep dt=1/sum(W(dt)) using root finding routine
ent_copy.f90	subroutine; copies ent_♯.dat to ent2_♯.dat
erf.f90	function; calculates errorfunction
fileinit.f90	subroutine; reads saved simulation variables to continue from the save point
F_plot.f90	subroutine; plots free energy
gammln.f90	function used for bico2.f90 from Numerical Recipes
gasdev.f90	function; calculates Gaussian distribution
get_r_cm.f90	function; calculates center of mass position of the chain
initiation.f90	subroutine; creates initial ensemble of chains
Lcount.f90	function; calculates primitive-path of the chain
life_time_set.f90	function; calculates characteristic life time, τ^i_{CD}, for entanglement
L_plot.f90	subroutine; plots primitive-path distribution
N_dist.f90	subroutine; generates N_i for initial ensemble
N_plot.f90	subroutine; plots N distribution
plot.f90	subroutine; triggers which plot to make
Q_dist.f90	subroutine; generates Q_i for initial ensemble
Q_plot.f90	subroutine; plots Q distribution
ranils.f	subroutine; starts initial sequence for random number generator
ranuls.f	function; random number generator from 0 (not included) to 1 (not included)
root.f90	function; finds root
sort.f90	subroutine; sorts list of numbers
taucopy.f90	subroutine; copies tau_♯.dat to tau2_♯.dat
tauwrite.f90	subroutine; writes stress tensors in tau_♯.dat
tstep.f90	subroutine; makes one τ_k step for all chains
visc_plot.f90	subroutine; plots elongational viscosity
visc_shear_plot.f90	subroutine; plots transient shear viscosity
w_bd.f90	subroutine; calculates all W^i_{SD} for one chain
w_cr.f90	subroutine; calculates all W^i_{CD} for one chain
z_dist.f90	subroutine; generates Z for initial ensemble
z_plot.f90	subroutine; plots Z distribution
Gcalc.f90	subroutine; calculates autocorrelation function G(t) using PCS matrix
Load.f90	subroutine; gives stress values to the PCS matrix
PCS.f90	subroutine; loads stress values in the PCS matrix
ReadCalc.f90	subroutine; reads PCS matrix when continue from save point
data.f90	module; carries global variables for PCS method code

results with the standard Intel compiler -O2 option, which is a different flag that can be used to optimize the program for speed.

2.2.4 To Run the Code

To run the program, copy input.dat, slm_run to your work directory and submit the script. The executable line is ./slm_run ♯, where the symbol, ♯, is an integer value that acts as a seed for a random number generator. For example, ./slm_run 1.

One simulation run corresponds to modeling one chain, with a large number of possible conformations. If the run is long enough, it is sufficient to describe a system composed of linear polymers. In order to achieve better statistics, it is desirable to run several runs with various seeds. Simulations should especially be performed from multiple random conformations when modeling network polymers.

2.2.5 Analyzing the Results

After each run, an output file of the relaxation modulus called G_♯.dat is produced, where ♯ is the same value that is use in ./slm_run ♯ (for example, G_1.dat from slm_run 1). If several output were produced using various seed values, numerous files consisting of the relaxation modulus are available and can be averaged to get a statistical average.

To average the relaxation modulus and create a Gnuplot (*1*) script file from the different runs (via various seed values), there is a program called gplot_avg.f90.

To compile this code, use the following command **ifort gplot_avg.f90 -o gplot_avg**, where the results are averaged by running the executable (./gplot_avg).

The averaged relaxation modulus data is saved into the file named G.dat, and the Gnuplot script is called Gplot.plt. The G.dat file contains the dimensionless time, t/τ_k, and the dimensionless relaxation modulus, $G(t)/(nk_BT)$ or $G(t)M_w/(\rho RT)$, where n is the number of chains per volume, k_B is the Boltzmann constant, T is the absolute temperature, M_w is the molecular weight, ρ is the polymer mass density, and R is the ideal gas constant. Loading the latter file in Gnuplot via the command, **load Gplot.plt**, plots the data in G.dat.

2.2.6 Calculating the Storage and Loss Modulus from the Relaxation Modulus

A Mathematica (*2*) program is provided to calculate the storage and loss modulus from the relaxation modulus obtained from the DSM program. This program is provided in the file called mathematica.zip, which is located in the same directory as the DSM program. Each

step in the file is explained in this section, so please study the Mathematica file with this guide.

To run the Mathematica program, it is necessary to export the relaxation modulus $G(t)$ produced from the DSM program. It is not necessary for the data to be evenly spaced in time. For the program to locate the data file, one must change the work directory path specified in the Mathematica file to the path where the data files are stored. After the relaxation modulus is imported, the Mathematica program allows for any data points associated with noise to be cut. From this simulation data, the plateau modulus G_N^0 is taken to be $G(0)$, which is the adjusted value briefly mentioned in section 2.1.2.

After this, the continuous spectrum of the relaxation times, $h(\tau)$, and the continuous relaxation modulus are calculated using equations 7 and 6, respectively. The continuous relaxation modulus is then compared to our simulated relaxation modulus, where weighted residules between the two relaxation modulus are calculated using a nonlinear least-squares algorithm to obtain optimal parameters for the power-law exponentials and time constants, α_i and τ_i, respectively. In the Mathematica file, the continuous relaxation modulus is calculated for one and two modes in the spectrum ($m = 1$ and $m = 2$). These two curves are then plotted against the simulated $G(t)$ to determine if the curve can adequately reproduce the profile. If the curves generated using only one or two modes in the spectrum do not reproduce the general shape of the simulated relaxation modulus, more modes should be used. Also in the nonlinear least-squares algorithm, which is used to determine the optimal parameters, α_i and τ_i, test values within an arbitary user defined range are specified. If the optimal parameters push the boundaries, the range for that variable should be expanded.

Using these optimal parameters, the relaxation modulus in the frequency domain, $G^*(\omega)$, is calculated using equations 8 and 9 for the storage $G'(\omega)$ and loss $G''(\omega)$ modulus, respectively, since $G^*(\omega) = G'(\omega) + iG''(\omega)$. The plateau modulus, G_N^0, appears in both equations 8 and 9, and since the relaxation modulus data from the DSM model is in units of nkT or $\rho RT/M_w$, it is necessary to multiply the plateau modulus by this factor since it was set equal to $G_N^0 = G(0)$ at the beginning of the Mathematica program.

The Mathematica file then reads in the experimental data files for the storage and the loss modulus located in the file path specified in the beginning of the program. The experimental data and the calculated modulus are then plotted for comparison purposes. The calculated storage and loss modulus are plotted and represented by a blue line, and the experimental storage and loss modulus are represented by dots. Different colors are used for different data sets. The experimental data and the simulated values of the moduli should not overlap at this point. To overlay the data, the shift parameters still need to be specified.

The first shift parameter, τ_k, is fitted through a single viscoelastic experiment. This value is used to shift the calculated modulus in the frequency domain so that it matches with the experimental data, and the user should modify its value until the simulated and experimental moduli overlap. Since the time dependence is normalized by τ_k, this parameter accompanies the frequency, ω, in equations 8 and 9 such that $\omega = \tau_k \omega$.

Since values of G_N^0 vary by 20% between different research laboratories due to a very strong dependence on the radius of the sample in oscillatory measurements, slight shifts in the calculated modulus are also allowed. This shift is allowed by multiplying the calculated $G'(\omega)$ and $G''(\omega)$ by a shift parameter, b, which shifts the data in the modulus domain.

3. Example Case 1: Storage and Loss Modulus for Polystyrene

We used the discrete slip-link model to calculate the storage and loss modulus for two linear polystyrene melts with molecular weights of 102 and 390 kg/mol, denoted as PS102 and PS390, respectively. Experimental data are provided by Nielsen et al. in reference 8. The data and input files associated with this example can be found in the file "mathematica.zip" along with the example Mathematica file described in section 2.2.6.

3.1 Experimental Data

To calculate the storage and loss modulus, numerous experimental parameters are necessary including the molecular weight of the polymer and a Kuhn length, M_w and M_k, respectively; polymer density, ρ; plateau modulus, G_N^0; and temperature, T. The data for this system are presented in table 2. The polydispersity index, M_w/M_n, is also given as a measure of the quality of the melt.

The two input parameters for the DSM model are the number of Kuhn segments, N_k, and the model parameter, β, which can be calculated using equations 10 and 12, respectively. The values for these two parameters are shown in table 3.

Table 2. Experimental data for polystyrene.

Name	M_w (kg/mol)	M_k (kg/mol)	M_w/M_n	ρ (g/cm^3)	T (K)	G_N^0 (kPa)
PS102	102	0.720	1.02	0.969	403.15	250
PS390	390	0.720	1.06	0.969	403.15	250
Reference	(*8*)	(*4*)	(*8*)	(*4*)	(*8*)	(*8*)

Table 3. Parameters needed for DSM program.

Name	N_k	β
PS102	143	17
PS390	542	17

3.2 DSM Program

Now that the parameters for the DSM program have been calculated, they are inputted into the input.dat file, where we consider both the constraint and sliding dynamics. For example, for PS102, this gives us the following input.dat:

```
17.d0                                  !β (double)
143                                    !N_k: number of Kuhn steps (int)
1                                      !Number of chains (int)
0.d0  0.d0  0.d0  0.d0  0.d0  0.d0     !deformation tensor (xx,xy,xz,yy,yz,zz) (double)
1                                      !CD off(0)/on monodisperse(1)/on from file(2) (int)
1                                      !SD off(0)/on (int)
1                                      !G(t) estimation on/off (int)
0                                      !f_d(t) and destruction rate estimation on/off (int)
0                                      !Diffusion on/off (int)
1.d0                                   !time step (double)
5000000.d0                             !T - simulation time
500000.d0                              !saving interval period
```

We then ran the DSM program for 10 initial conditions using seeds 1–10, which produced 10 files of data for the relaxation modulus, $G(t)$. To average the results, we then used the program gplot_avg.f90, as described in section 2.2.5.

The averaged relaxation modulus data we obtained from these 10 seeds can be found in table 4 for PS102 and PS390, respectively, where Gnuplot can be used to plot the data using the scripts Gplot.plt, as seen in figures 2 and 3.

Table 4. Simulated relaxation modulus data.

t/τ_k	$G(t)/(nkT)$ (PS102)	$G(t)/(nkT)$ (PS390)
0.000000000000000E+000	6.91083309464347	29.8898725860124
1.00000000000000	6.19796426384710	28.0854137134369
2.00000000000000	5.95983031951911	27.5037174001375
3.00000000000000	5.79844528724713	27.1185525580062
4.00000000000000	5.69224450413905	26.8392570251266
6.00000000000000	5.49800457660264	26.3922005619445
8.00000000000000	5.35714272394803	26.0744408076688
12.0000000000000	5.13025904543506	25.5715868220268
16.0000000000000	4.96724710835751	25.2209373684510
24.0000000000000	4.70657510398101	24.6714678139358
32.0000000000000	4.52100731226043	24.2937151609842
48.0000000000000	4.22627717740879	23.7064962479896
64.0000000000000	4.01801208834300	23.3074541480095
96.0000000000000	3.68833428551522	22.6907666578126
128.000000000000	3.45814843502180	22.2730552608380
192.000000000000	3.09512842906176	21.6298887133200
256.000000000000	2.84306683982082	21.1969638257405
384.000000000000	2.44513902693429	20.5218935407985
512.000000000000	2.17567922785316	20.0622219649911
768.000000000000	1.76410767550662	19.3435630679622
1024.00000000000	1.48925114318622	18.8506897264010
1536.00000000000	1.08002363664520	18.0739321422468
2048.00000000000	0.824844929292372	17.5354852799470
3072.00000000000	0.485397404135916	16.6597892714719
4096.00000000000	0.325533615384035	16.0468021973291
6144.00000000000	0.140749736670129	15.0892088120946
8192.00000000000	7.491837480645815E-002	14.4165194076411
12288.0000000000	1.146546863200251E-002	13.3450624449661
16384.0000000000	1.636253498597255E-003	12.5931522095167
24576.0000000000	-5.738856952541065E-003	11.4093068236647
32768.0000000000	-5.747403926758270E-004	10.6004916011373
49152.0000000000	4.052422628952885E-003	9.29707756201879
65536.0000000000	1.033559509771989E-002	8.31446807859437
98304.0000000000	3.084434564843599E-003	6.90482027244776
131072.000000000	4.707060633778353E-003	5.89862351214386
196608.000000000	-4.991878704392457E-003	4.42668804312963
262144.000000000	9.252570356156216E-004	3.42334979090572
393216.000000000	1.776842640453745E-003	2.20392772330922
524288.000000000	1.627208379936440E-004	1.72105885610507
786432.000000000	8.862902171381414E-004	0.718130734528209
1048576.00000000	-6.850332688539823E-004	0.195794718887397
1572864.00000000	2.111278482574322E-003	-0.340805533589627
2097152.00000000		-0.114542256332998
3145728.00000000		-4.146371847763596E-002

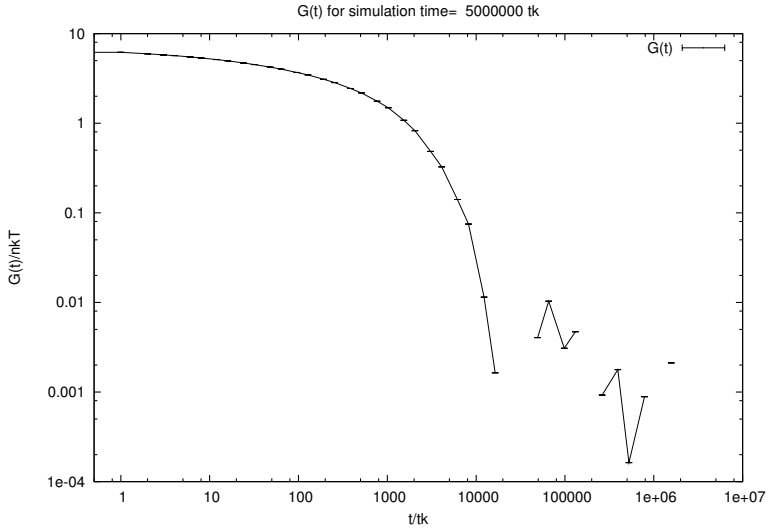

Figure 2. The simulated relaxation modulus produced using Gplot.plt in Gnuplot for PS102.

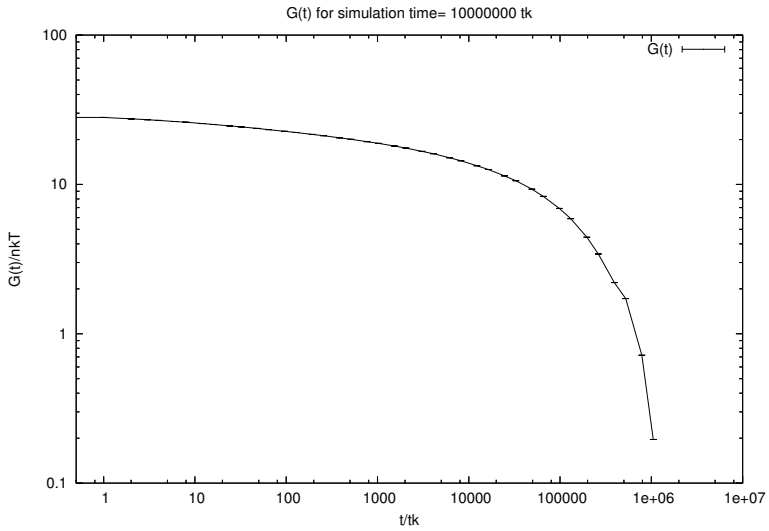

Figure 3. The simulated relaxation modulus produced using Gplot.plt in Gnuplot for PS390.

3.3 Mathematica

To calculate the storage and loss modulus from the simulated relaxation data, we use the Mathematica file, as described in section 2.2.6. We compare the results to the experimental values of the modulus presented in reference 8. This experimental data are presented in tables 5, 6, 7, and 8.

Table 5. Experimental storage modulus data for the PS102 from reference 8.

ω (rad/s)	$G'(\omega)$ (kPa)
5.05825E-4	1.73876E-1
1.09648E-3	7.12612E-1
2.29087E-3	2.96123E+0
5.15229E-3	1.19696E+1
1.07647E-2	3.61875E+1
2.37684E-2	6.93172E+1
5.15229E-2	1.03518E+2
8.16582E-2	1.23906E+2
1.29420E-1	1.44264E+2
2.01372E-1	1.63385E+2
3.13329E-1	1.87617E+2
5.05825E-1	2.03849E+2
8.01678E-1	2.30868E+2
1.29420E+0	2.68800E+2
2.05116E+0	3.00246E+2
3.13329E+0	3.30765E+2
4.96592E+0	3.90474E+2
8.01678E+0	4.42227E+2
1.24738E+1	5.51749E+2
2.01372E+1	6.42403E+2
3.13329E+1	7.79636E+2
5.05825E+1	9.72720E+2

Table 6. Experimental loss modulus data for the PS102 from reference 8.

ω (rad/s)	$G''(\omega)$ (kPa)
4.99812E-4	4.16719E+0
1.10904E-3	8.52373E+0
2.41598E-3	1.79228E+1
4.97794E-3	3.46973E+1
1.10367E-2	5.46147E+1
2.39981E-2	6.26056E+1
5.12192E-2	6.88599E+1
8.13123E-2	7.07292E+1
1.31517E-1	7.67662E+1
2.04999E-1	8.33237E+1
3.31585E-1	9.16912E+1
5.26626E-1	1.08104E+2
8.21076E-1	1.27458E+2
1.35332E+0	1.56609E+2
2.11053E+0	2.00573E+2
2.58661E+0	2.06109E+2
3.23059E+0	2.43102E+2
4.10913E+0	2.63960E+2
5.03796E+0	3.07082E+2
6.29144E+0	3.47522E+2
8.30407E+0	3.82554E+2
1.03728E+1	4.70271E+2
1.27170E+1	5.39606E+2
1.61780E+1	6.19126E+2
2.09658E+1	7.20203E+2
2.61856E+1	8.49467E+2
3.21033E+1	9.74710E+2
4.23785E+1	1.11828E+3
5.10105E+1	1.33739E+3

Table 7. Experimental storage modulus data for the PS390 from reference 8.

ω (rad/s)	$G'(\omega)$ (kPa)
5.24807E-5	1.64519E+1
9.46237E-5	3.28486E+1
1.70608E-4	5.25679E+1
2.33346E-4	6.46861E+1
2.96483E-4	7.22535E+1
4.20727E-4	8.52964E+1
5.34564E-4	9.26759E+1
7.31139E-4	1.03518E+2
9.12011E-4	1.06421E+2
1.31826E-3	1.22204E+2
1.64437E-3	1.20526E+2
2.29087E-3	1.38402E+2
2.91072E-3	1.40329E+2
4.13048E-3	1.50376E+2
5.24807E-3	1.52470E+2
7.31139E-3	1.61141E+2
8.95365E-3	1.63385E+2
1.05682E-2	1.75083E+2
1.31826E-2	1.70307E+2
1.64437E-2	1.82499E+2
1.87068E-2	1.87617E+2
2.29087E-2	1.85041E+2
2.85759E-2	1.90230E+2
3.25087E-2	2.01050E+2
4.05509E-2	1.95565E+2
5.15229E-2	2.03849E+2
5.75440E-2	2.09566E+2
7.31139E-2	2.03849E+2
9.28966E-2	2.15443E+2
1.05682E-1	2.27697E+2
1.34276E-1	2.15443E+2
1.64437E-1	2.21486E+2
2.29087E-1	2.24570E+2
3.37287E-1	2.50842E+2
4.05509E-1	2.34083E+2
5.54626E-1	2.68800E+2
7.17794E-1	2.57876E+2
1.00000E+0	3.12965E+2
1.27057E+0	2.80187E+2

Table 8. Experimental loss modulus data for the PS390 from reference 8.

ω (rad/s)	$G''(\omega)$ (kPa)
5.17743E-5	3.13173E+1
9.02209E-5	4.12223E+1
1.66111E-4	4.72671E+1
2.36004E-4	4.72387E+1
2.94612E-4	4.72208E+1
3.95995E-4	4.71969E+1
5.32199E-4	4.52616E+1
7.42282E-4	4.52358E+1
9.43900E-4	4.52172E+1
1.31650E-3	4.51915E+1
1.61312E-3	4.33453E+1
2.29129E-3	3.98798E+1
2.91377E-3	4.04168E+1
4.21647E-3	3.87546E+1
5.46132E-3	3.76838E+1
7.47605E-3	3.51545E+1
9.16245E-3	3.61249E+1
1.02358E-2	3.46545E+1
1.32577E-2	3.36970E+1
1.71733E-2	3.36821E+1
1.88377E-2	3.46184E+1
2.43971E-2	3.27463E+1
3.16027E-2	3.27318E+1
4.24797E-2	3.31695E+1
5.92660E-2	3.65096E+1
7.26163E-2	3.45386E+1
9.23638E-2	3.75019E+1
1.05162E-1	4.24473E+1
1.36175E-1	3.79974E+1
1.60869E-1	4.18356E+1
1.86562E-1	4.60630E+1
2.28625E-1	4.60470E+1
3.37313E-1	5.89790E+1
4.36973E-1	6.06012E+1
5.87845E-1	7.98034E+1
7.20324E-1	7.76059E+1
1.04356E+0	1.07977E+2
1.27886E+0	1.07940E+2

We use PS102 to calculate the parameter, τ_k, which can also be used in the calculation for PS390, since the parameter is only dependent on the chemistry and temperature of the polymer melt.

After importing and plotting the simulated relaxation data, we notice that there are some data points that are associated with noise (see figure 2). In the Mathematica file, there is a variable "a", which indicates how many points will be cut from the simulated data. For this system this value should be set to 12. After removing the noise, we obtain the following plot of the relaxation modulus in figure 4.

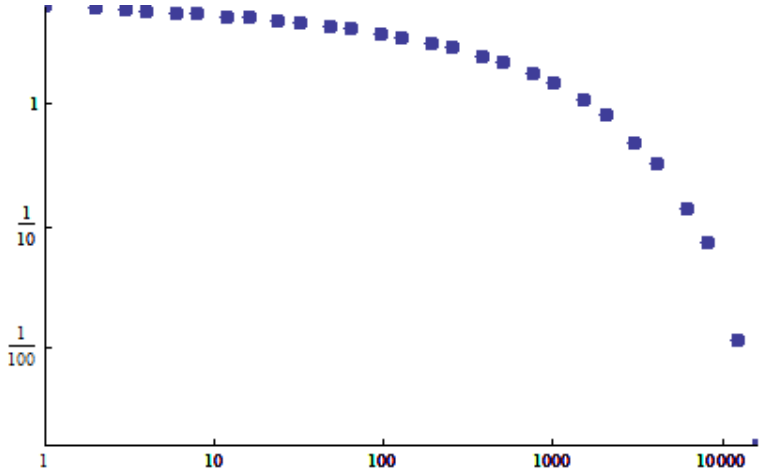

Figure 4. The simulated relaxation modulus $G(t)/(nk_BT)$ vs. t/τ_k for PS102.

The relaxation times $h(\tau)$ and the continuous relaxation modulus are then calculated (equations 7 and 6), where we consider one and two modes in the spectrum ($m = 1$ and $m = 2$). When one mode is considered, the following parameters for the power-law exponentials and time constants, $\alpha_0 = 0.227$, $\alpha_1 = 0.227$, $\tau_0 = 0.075$, $\tau_1 = 3952.4$, are obtained through the nonlinear least-squares algorithms. For two modes, we obtain the following optimal parameters: $\alpha_0 = 0.1996$, $\alpha_1 = 0.1996$, $\alpha_2 = 9.99$, $\tau_0 = 0.164$, $\tau_1 = 2511.99$, and $\tau_2 = 3066.87$.

The two fits of the continuous relaxation modulus are then plotted in figure 5, where the red and black lines are produced using one or two modes, respectively.

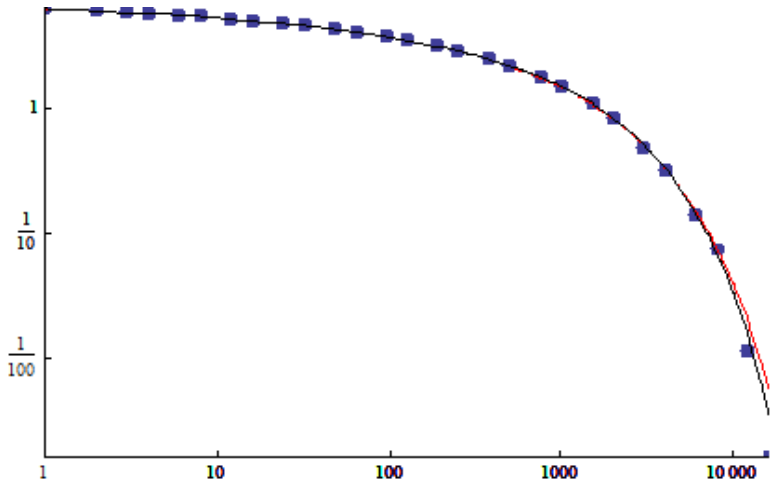

Figure 5. The simulated relaxation modulus (data points), $G(t)/(nk_BT)$ vs. t/τ_k , fitted with a continuous line where one (red) and two (black) modes are used in equations 7 and 6 for PS102.

Using these fit parameters, we calculated the Fourier transform of continuous relaxation modulus, which is then used to calculate the storage and loss modulus. Calculating $G^*(\omega)$ involves a multiplication by the plateau modulus, G_N^0. Up till now, the plateau modulus was normalized by nkT, so we must multiply this value by nkT or $\rho RT/M_w$, in units of kPa, before applying it in the calculation of the modulus. Comparing the calculated loss and storage modulus to the experimental data, the values of τ_k and b are calculated, which will shift the data in the frequency (ω) and modulus domains, respectively. For a shift parameter of $\tau_k = 0.03s$ and $b = 1.6$, an adequate fit of the data is produced, as seen in figure 6. The shift parameter, b, is due to experimental fluctuations, and its value can vary from experiment to experiment, so this value can be different for PS390. In this figure, the circles represent the experimental data and the lines represent the calculated modulus, where orange/green and red/blue represents the storage and loss modulus, respectively. Since the DSM does not describe the glassy regime in polymer dynamics, which dominates at high frequencies, the model prediction deviates from the experimental data at high frequency.

Now that we have calculated τ_k from PS102, we can use it in the calculation for PS390. First, the Mathematica file imports the simulated relaxation modulus for PS390 produced from the DSM. The data lacks any noise, so the truncation parameter a is set equal to zero. Figure 7 shows the relaxation modulus produced by Mathematica for PS390.

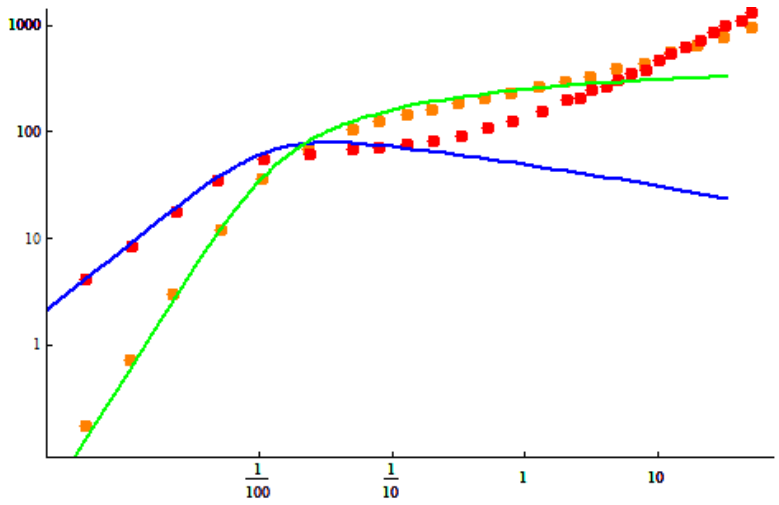

Figure 6. A comparison of the experimental and calculated modulus, in units of kPa, as a function of frequency (ω (rad/s)) for PS102. The data points represent experimental data points (orange = storage modulus, red = loss modulus). The continuous lines represent the calculated values (green = storage modulus, blue = loss modulus).

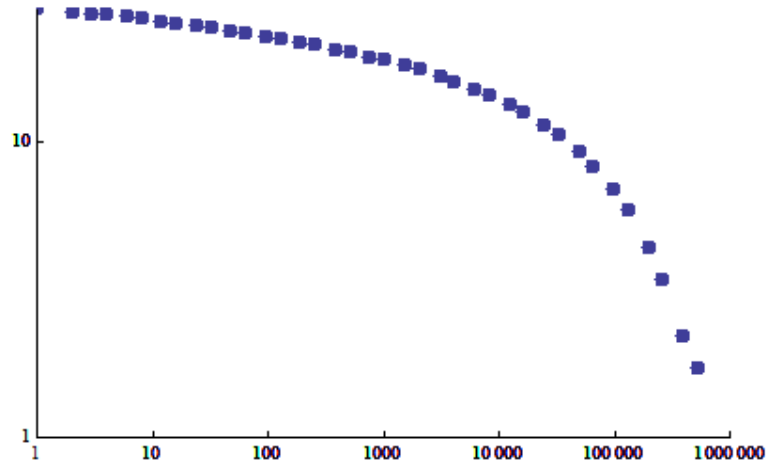

Figure 7. The simulated relaxation modulus $G(t)/(nk_BT)$ vs. t/τ_k for PS390.

The relaxation times $h(\tau)$ and the continuous relaxation modulus are again calculated using equations 7 and 6 for one and two modes in the spectrum. When one mode is considered, the following parameters are obtained for the power-law exponentials and time constants, $\alpha_0 = 0.172$, $\alpha_1 = 0.172$, $\tau_0 = 0.00078$, $\tau_1 = 651360$, while for two modes we obtain the following constants: $\alpha_0 = 0.172$, $\alpha_1 = 0.172$, $\alpha_2 = 200$, $\tau_0 = 0.000784$, $\tau_1 = 654616$, $\tau_2 = 108549$. The fit of the simulated data is presented in figure 8 for both cases. Now that we have the fit parameters, we then calculate the Fourier transform of the continuous relaxation modulus, which we then use to calculate the storage and loss modulus. Again, we multiply the plateau modulus by $\rho RT/M_w$ before applying it to this calculation. When comparing the simulated moduli with the experimental data, we use the value of $\tau_k = 0.03s$ calculated in the PS102. This value adequately shifts the data in the frequency domain. We then shift the data in the modulus domain with a shift parameter, $b = 1.15$. A comparison of the simulated and experimental data can be seen in figure 9, where the lines and circle represent the calculated modulus and experimental data, respectively.

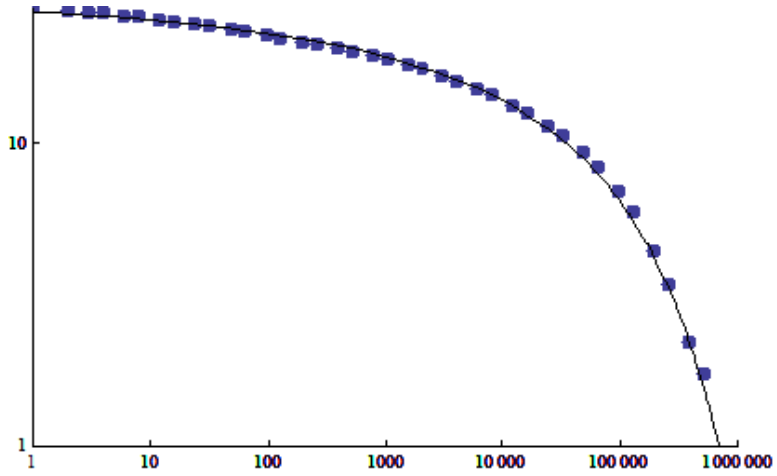

Figure 8. The simulated relaxation modulus (data points), $G(t)/(nk_BT)$ vs. t/τ_k , fitted with a continuous line where one (red) and two (black) modes are used in equations 7 and 6 for PS390.

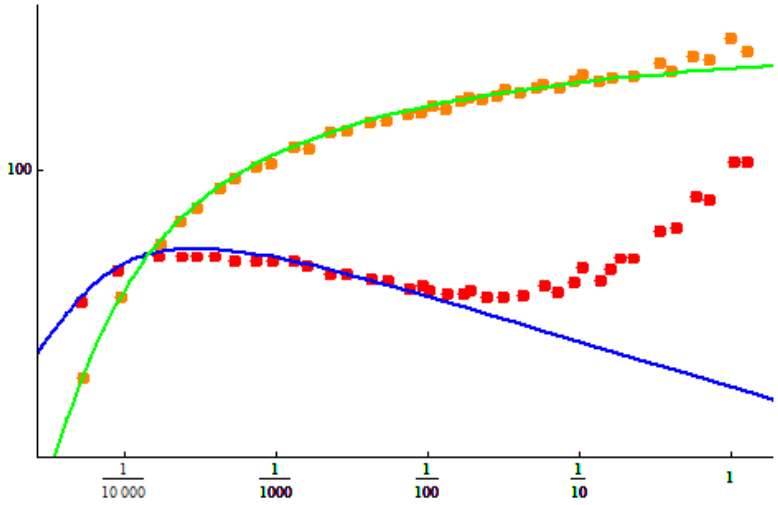

Figure 9. A comparison of the experimental and calculated modulus, in units of kPa, as a function of frequency (ω (rad/s)) for PS390. The data points represent experimental data points (orange = storage modulus, red = loss modulus). The continuous lines represent the calculated values (green = storage modulus, blue = loss modulus).

4. Conclusion

Step by step, we illustrated how to use the DSM code provided by Professor Schieber to calculate the storage and loss modulus for two different molecular weights of polystyrene. This code is composed of a FORTRAN 90 code, which is used to calculate the relaxation modulus. A Mathematica code is then used to compute the storage and loss modulus from the relaxation modulus. To parameterize the fitting parameters in the model, experimental data from one linear viscoelastic experiment are necessary although current research is being conducted to circumvent this necessity. In our test case, we observed good agreement between the simulated and experimental results in the elastic regime. This theory has also been applied to other homopolymer solutions and melts with similar agreement.

References

[1] Gnuplot homepage. http://gnuplot.info (accessed August 25, 2010).

[2] Mathematica homepage. http://www.wolfram.com (accessed August 25, 2010).

[3] Doi, M.; Edwards S. Dynamics of concentrated polymer systems. Part 2.Molecular motion under flow. *Journal of the Chemical Society, Faraday Transactions 2: Molecular and Chemical Physics* **1978**, *74*, 1802.

[4] Fetters, L.; Lohse, D.; Colby, R. *Physical Properties of Polymer Handbook*; AIP Press, 447, 2007.

[5] Khaliullin, R. N.; Schieber, J. D. Analytic Expressions for the Statistics of the Primitive-Path Length in Entangled Polymers. *Physical Review Letters* **2007**, *100*, 188302.

[6] Khaliullin, R. N.; Schieber, J. D. Self-Consistent Modeling of Constraint Release in a Single-Chain Mean-Field Slip-Link Model. *Macromolecules* **2009**, *42* (19), 7504.

[7] Nair, D. M.; Schieber, J. D. Linear Viscoelastic Predictions of a Consistently Unconstrained Brownian Slip-link Model. *Macromolecules* **1996**, *39*, 3386.

[8] Nielsen, J. K.; Rasmussen, H. K.; Hassager, O.; McKinley, G. H. Elongational viscosity of monodisperse and bidisperse polystyrene melts. *Journal of Rheology* **2006**, *50*, 453.

[9] Schieber, J. D.; Neergaard J.; Gupta S. A full-chain, temporary network model with sliplinks, chain-length fluctuations, chain connectivity and chain stretching. *Journal of Rheology* **2003**, *47*, 213.

[10] Schieber, J. D.; Nair, D. M.; Kitkrailard, T. Comprehensive Comparisons with Nonlinear Flow Data of a Consistently Unconstrained Brownian Slip-link Model. *Journal of Rheology* **2007**, *51*, 1111.

NO. OF COPIES	ORGANIZATION
1 ELEC	ADMNSTR DEFNS TECHL INFO CTR ATTN DTIC OCP 8725 JOHN J KINGMAN RD STE 0944 FT BELVOIR VA 22060-6218
1 CD	OFC OF THE SECY OF DEFNS ATTN ODDRE (R&AT) THE PENTAGON WASHINGTON DC 20301-3080
1	US ARMY RSRCH DEV AND ENGRG CMND ARMAMENT RSRCH DEV & ENGRG CTR ARMAMENT ENGRG & TECHNLGY CTR ATTN AMSRD AAR AEF T J MATTS BLDG 305 ABERDEEN PROVING GROUND MD 21005-5001
1	PM TIMS, PROFILER (MMS-P) AN/TMQ-52 ATTN B GRIFFIES BUILDING 563 FT MONMOUTH NJ 07703
1	US ARMY INFO SYS ENGRG CMND ATTN AMSEL IE TD A RIVERA FT HUACHUCA AZ 85613-5300
1	COMMANDER US ARMY RDECOM ATTN AMSRD AMR W C MCCORKLE 5400 FOWLER RD REDSTONE ARSENAL AL 35898-5000
1	US GOVERNMENT PRINT OFF DEPOSITORY RECEIVING SECTION ATTN MAIL STOP IDAD J TATE 732 NORTH CAPITOL ST NW WASHINGTON DC 20402
1	US ARMY RSRCH LAB ATTN RDRL WMM G J ANDZELM BLDG 4600 RM C204 ABERDEEN PROVING GROUND MD 21005

NO. OF COPIES	ORGANIZATION
1	US ARMY RSRCH LAB ATTN RDRL WMM G T CHANTAWANSRI BLDG 4600 RM C228 ABERDEEN PROVING GROUND MD 21005
1	US ARMY RSRCH LAB ATTN RDRL WMM G Y SLIOZBERG BLDG 4600 RM C213 ABERDEEN PROVING GROUND MD 21005
1	US ARMY RSRCH LAB ATTN RDRL CIM G T LANDFRIED BLDG 4600 ABERDEEN PROVING GROUND MD 21005-5066
3	US ARMY RSRCH LAB ATTN IMNE ALC HRR MAIL & RECORDS MGMT ATTN RDRL CIM L TECHL LIB ATTN RDRL CIM P TECHL PUB ADELPHI MD 20783-1197

TOTAL: 14 (1 ELEC, 1 CD, 12 HCS)

INTENTIONALLY LEFT BLANK.